UCID- 18825

A New Formulation of Backward Differenced Linear Multistep Methods for Ordinary Differential Equations

Alan M. Winslow

October 2, 1980

Lawrence Livermore Laboratory

Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore Laboratory under Contract W-7405-Eng-48.

A New Formulation of Backward Differenced Linear Multistep Methods for Ordinary Differential Equations

Alan M. Winslow

Consider the ordinary differential equation

$$\dot{y} = \frac{dy}{dt} = f(y,t) \tag{1}$$

to be integrated numerically by a k^{th} order linear multistep method. Choosing backward differencing for application to stiff equations[1] we have

$${}_k\dot{y}^{n+1} = \dot{y}(\text{lin. comb. of } y^{n+1}, y^n, \cdot \cdot \cdot , y^{n-k+1}) = f(y^{n+1}, t^{n+1}) \tag{2}$$

where t^{n-k+1}, t^{n-k+2}, . . . , t^{n+1} represent successive times, not necessarily equally spaced, and $1 \leq k \leq K$, where usually K = 5 or 6.

The novel feature of our formulation is the manner in which we store the "history" array $y^n, y^{n-1}, \ldots, y^{n-k+1}$ needed for the left hand side of (2). We use the extrapolants ${}_jy^{n+1}$ $(j = 0, 1, \cdot \cdot \cdot, k)$ each obtained by fitting a j^{th} degree polynomial to the $j + 1$ points $\left\{y^{n-1}\right\}_0^j$ and evaluated at $t = t^{n+1}$. These extrapolants are a natural choice because they form a sequence of successive approximations to y^{n+1}.

We begin our derivation with the usual form of the Newton interpolation polynomial ${}_ky^o(t)$ of k^{th} degree[2] for the function y(t) over the set $\left\{t_i\right\}_o^k$ evaluated at t:

$$\begin{aligned} y(t) &= y[t_o] + (t - t_o)\, y[t_o, t_1] + (t - t_o)(t - t_1)\, y[t_o, t_1, t_2] \\ &\quad + \cdot \cdot \cdot \cdot + (t - t_o)(t - t_1) \cdot \cdot \cdot (t - t_{k-1})\, y[t_o, t_1, \ldots, t_k] \\ &\quad + {}_ke^o(t) \\ &= {}_ky^o(t) + {}_ke^o(t) \end{aligned} \tag{3}$$

Here $y[t_o, \cdots, t_j]$ represents the j^{th} divided difference which satisfies the recursion relation

$$y[t_o] = y(t_o)$$

$$y[t_o, \cdots, t_n] = \frac{y[t_o, \cdots, t_{n-1}] - y[t_1, \cdots, t_n]}{t_o - t_n}$$

The error term evaluated at t is given by[2]

$${}_k e^o(t) = (t - t_o)(t - t_1) \cdots (t - t_k)\, y[t_o, t_1, \cdots, t_k, t]$$

Dropping ${}_k e$ temporarily, we write the next higher interpolation polynomial over $\{t_i\}_o^{k+1}$ evaluated at t:

$${}_{k+1}y^o(t) = {}_k y^o(t) + (t - t_o)(t - t_1) \cdots (t - t_k)\, y[t_o, t_1, \cdots, t_{k+1}] \quad (4)$$

Shifting the evaluation point to t_o and the base set to $\{t_i\}_1^{k+1}$, Eq. (3) becomes

$$y(t_o) = {}_k y^1(t_o) + {}_k e^1(t_o) \quad (5)$$

where

$${}_k e^1(t_o) = (t_o - t_1)(t_o - t_2) \cdots (t_o - t_{k+1})\, y[t_1, t_2, \cdots, t_{k+1}, t_o]$$

Using (5), we can rewrite (4) in the form

$${}_{k+1}y\,(t) = {}_k y^o(t) + \frac{(t - t_o)(t - t_1) \cdots (t - t_k)}{(t_o - t_1)(t_o - t_2) \cdots (t_o - t_{k+1})}[y(t_o) - {}_k y^1(t_o)] \quad (6)$$

where we have made use of the symmetry property of divided differences in permuting their arguments.

Letting $t = t^{n+1}$, $t_i = t^{n-i}$ $(i = 0, 1, \cdots, k+1)$, $y(t^n) = y^n$, ${}_k y^o(t) = {}_k y^{n+1}$, and ${}_k y^1(t_o) = {}_k y^n$ we have

$$ {}_{k+1}y^{n+1} = {}_{k}y^{n+1} + \frac{(t^{n+1} - t^{n})(t^{n+1} - t^{n-1}) \cdots (t^{n+1} - t^{n-k})}{(t^{n} - t^{n-1})(t^{n} - t^{n-2}) \cdots (t^{n} - t^{n-k-1})} \left(y^{n} - {}_{k}y^{n} \right) \quad (7) $$

$(k = 0, 1, \cdots, K)$, where ${}_{o}y^{n+1} = y^{n}$. From the solution y^{n} and the extrapolants ${}_{k}y^{n}$ at $t = t^{n}$, we use (7) to form the successive extrapolants ${}_{k}y^{n+1}$ for $t = t^{n+1}$.

We make use of (7) to rewrite the Newton interpolation polynomial (3) in extrapolant form. Using the set $\left\{ y^{n+1-i} \right\}_{o}^{k}$ and evaluating at t, we have

$$ {}_{k}y^{n+1}(t) = y^{n+1} + \frac{t-t^{n+1}}{t^{n+1}-t^{n}} (y^{n+1}-y^{n}) + \frac{(t-t^{n+1})(t-t^{n})}{(t^{n+1}-t^{n})(t^{n+1}-t^{n-1})} (y^{n+1}-{}_{1}y^{n+1}) $$

$$ + \cdots + \frac{(t-t^{n+1})(t-t^{n}) \cdots (t-t^{n-k+2})}{(t^{n+1}-t^{n})(t^{n+1}-t^{n-1}) \cdots (t^{n+1}-t^{n-k+1})} (y^{n+1}-{}_{k-1}y^{n+1}), \quad (8) $$

and differentiating (8) we obtain our difference approximation for the lefthand side of (2):

$$ {}_{k}\dot{y}^{n+1} = \frac{d}{dt}\left[{}_{k}y^{n+1}(t) \right]_{t=t^{n+1}} = \frac{y^{n+1} - y^{n}}{\Delta t^{n+\frac{1}{2}}} + \frac{y^{n+1} - {}_{1}y^{n+1}}{\Delta t^{n+\frac{1}{2}} + \Delta t^{n-\frac{1}{2}}} \quad (9) $$

$$ + \frac{y^{n+1} - {}_{2}y^{n+1}}{\Delta t^{n+\frac{1}{2}} + \Delta t^{n-\frac{1}{2}} + \Delta t^{n-3/2}} + \cdots + \frac{y^{n+1} - {}_{k-1}y^{n+1}}{\Delta t^{n+\frac{1}{2}} + \cdots + \Delta t^{n+3/2-k}} $$

which is the desired linear combination of $\left\{ y^{n+1-i} \right\}_{o}^{k}$ indicated in (2). We insert (9) into (2) and solve for y^{n+1}, our k^{th} order approximation to $y(t^{n+1})$.

We adjust the time step and the order k in order to control the local truncation error

$$ {}_{k}\varepsilon^{n+1} \equiv y^{n+1} - y(t^{n+1}) $$

which, for a k^{th} order scheme, is of order $(\Delta t)^{k+1}$. For this purpose we make use of the interpolation error ${}_k e^{n+1}$, which, according to (5) is given by

$$ {}_k e^{n+1} = y^{n+1} - {}_k y^{n+1} $$

We can write ${}_k e^{n+1}$ in the form[3]

$$ {}_k e^{n+1} = (t^{n+1} - t^n)(t^{n+1} - t^{n-1}) \ldots (t^{n+1} - t^{n-k-1}) \frac{y^{(k+1)}(\xi)}{(k+1)!} $$

where $t^{n-k-1} \le \xi \le t^{n+1}$. For equal time steps Δt, we have

$$ {}_k e^{n+1} = (\Delta t)^{k+1} \, y^{(k+1)}(\xi) $$

showing that ${}_k e^{n+1}$ is also of order $(\Delta t)^{k+1}$. It can be shown[4] that $\left({}_k e^{n+1}\right)^{1/(k+1)}$ differs from $\left({}_k \varepsilon^{n+1}\right)^{1/(k+1)}$ only by a factor which lies between 1.5 and 2, which we shall omit.

Defining the fractional error

$$ \delta_k = \left| \frac{y^{n+1} - {}_k y^{n+1}}{y^{n+1}} \right| , $$

we use δ_k to control the time step and order in the following way. Comparing δ_k with some preassigned value δ_o, we calculate the new time step $\Delta t^{n+3/2}$ from

$$ \Delta t^{n+3/2} = (\delta_o/\delta_k)^{\frac{1}{k+1}} \, \Delta t^{n+\frac{1}{2}} \tag{10} $$

which increases or decreases the time step to try to maintain $\delta_k \approx \delta_o$. For continuous $f(y,t)$ this is usually sufficient, without having to repeat a time step. Changing Δt every time step in this manner has led to no stability problems. For order control we compare the numbers $(\delta_o/\delta_\ell)^{\frac{1}{\ell+1}}$ $(\ell = k, k \pm 1)$ and choose ℓ in order to maximize the new time step.

Another use of (8) is interpolation. To find the solution at some time t_p, $(t^n < t_p < t^{n+1})$, we interpolate to k^{th} order between y^n and y^{n+1} by using (8) with $t = t_p$.

An important application of Eq. (9) is to chemical kinetics, for which the righthand side of (1) takes the form

$$f(y,t) = -a(y,t)\, y + b(y,t) \tag{11}$$

Substituting (9) and (11) into (2) we get

$$y^{n+1} = \frac{S_k + b(y^{n+1},\, t^{n+1})\, \Delta t^{n+\frac{1}{2}}}{T_k + a(y^{n+1},\, t^{n+1})\, \Delta t^{n+\frac{1}{2}}} \tag{12}$$

where $S_k = \sum_{0}^{k-1} {}_j r_j y^{n+1}$, $T_k = \sum_{0}^{k-1} {}_j r$, ${}_j r = \dfrac{\Delta t^{n+\frac{1}{2}}}{\sum_{0}^{j} \Delta t^{n+\frac{1}{2}-i}}$. Notice that

$S_1 = y^n$, $T_1 = 1$. For any order we see from (12) that $\lim_{\Delta t \to \infty} y^{n+1} = b/a$ as it should, representing the quasisteady state solution $\dot{y} = 0$.

This method has been applied to the numerical solution of systems of chemical kinetics equations, having the form

$$\dot{y}_i = -a_i(y_1, \cdots, y_I, t)\, y_i + b(y_1, \cdots, y_I, t) \qquad (i = 1, 2, \cdots, I)$$

A predictor-corrector method using ${}_k y^{n+1}$ for the predictor and functional iteration based on (12) for the corrector has proved successful for the cases $k \leq 2$ (see Ref. 5). A new program with $k \leq 6$ is currently being tested.

The principal advantage of this formulation is its simplicity. The same extrapolants are used for prediction, correction, interpolation and control and are easily updated recursively. The chemical kinetics program, in fact, consists only of Eqs. (7), (10) and (12), although iteration by Newton's method could be used instead of (12). It is hoped to make it at least as efficient and accurate as the LLL (Hindmarsh) version of Gear's program.[6]

NOTE: This UCID is unchanged from the original UCIR-681 (January 17, 1973) except for the correction of minor errors.

A. M. Winslow
Oct. 2, 1980

References

1. C. F. Curtiss and J. O. Hirschfelder, Proc. Nat. Acad. Sci. 38, 235 (1952).

C. W. Gear, A.C.M. 14, No. 3, 176 (1971).

R. K. Brayton, F. G. Gustavson, and G. D. Hachtel, Proc. IEEE, Jan. 1972, p. 98.

(This last reference, which was brought to our attention after our work was completed, is also based on Newton's interpolation polynomial but uses a different history array.)

2. E. Isaacson and H. B. Keller, Analysis of Numerical Methods (John Wiley and Sons, 1966) Chapter 6, Sec. 1.

3. Ibid., Eq. (9).

4. A. C. Hindmarsh, private communication; see also UCID-30050, part 3 (1972).

5. UCRL-73911 (January, 1972).

6. A. C. Hindmarsh, UCID-30001 (1971; revised Sept., 1972).